

APPENDIX A.V

Source code file named translations.pl.



```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   File       : translations.pl
%   Primary Authors : David Martin, Adam Cheyer
%   Purpose    : Provides translations for backward compatibility with OAA 1.0
%
%   -----
%   Unpublished-rights reserved under the copyright laws of the United States.
%
%   -----
%   Unpublished Copyright (c) 1998, SRI International.
%   "Open Agent Architecture" and "OAA" are Trademarks of SRI International.
%   -----

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

% This file is loaded by facilitator code, and thus no
% module imports are needed here.

```

```

% Currently, we support a 3.0 facilitator with a mix of 3.0 and/or pre-3.0
% clients.
% A pre-3.0 facilitator with a 3.0 client is NOT supported, and probably
% never will be.

```

```

:- multifile oaa_AppDoEvent/2.

```

```

% At present we only support the case where the facilitator is 3.0, and
% the client is pre-3.0.

```

```

% Here we can ignore the languages.
oaa_event_translation(2.0, L1, 3.0, L2, Connection, Event1, Event2) :-
    oaa_event_translation(2.1, L1, 3.0, L2, Connection, Event1, Event2).
oaa_event_translation(2.1, _L1, 3.0, _L2, _Connection, Event1, Event2) :-
    ( Event1 = event(From, Contents1, Priority) ->
        Params2 = [from(From), priority(Priority)]
    | Event1 = event(From, Contents1) ->
        Params2 = [from(From)]
    | Event1 = Contents1 ->
        Params2 = []
    ),
    ( ev_trans_21_30(Contents1, Contents2) ->
        true
    | otherwise ->
        Contents2 = Contents1
    ),
    Event2 = event(Contents2, Params2).

```

```

% Here we can ignore the languages.
oaa_event_translation(3.0, L1, 2.0, L2, Connection, Event1, Event2) :-
    oaa_event_translation(3.0, L1, 2.1, L2, Connection, Event1, Event2).
oaa_event_translation(3.0, _L1, 2.1, _L2, _Connection, Event1, Event2) :-
    Event1 = event(Contents1, Params1),
    ( ev_trans_30_21(Contents1, Params1, Contents2) ->
        true
    | otherwise ->
        Contents1 = Contents2
    ),
    ( memberchk(from(KS), Params1) ->

```

```

        Event2 = event(KS, Contents2)
    | otherwise ->
        Event2 = Contents2
    ),
    !.
    % Anything not specified explicitly stays the same:
oaa_event_translation(3.0, _L1, 2.1, _L2, _Connection, E1, E1).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The following could go to or from the facilitator.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ev_trans_21_30(trace_on, ev_trace_on).
ev_trans_21_30(trace_off, ev_trace_off).
ev_trans_21_30(tcp_trace_on, ev_com_trace_on).
ev_trans_21_30(tcp_trace_off, ev_com_trace_off).
ev_trans_21_30(debug_on, ev_debug_on).
ev_trans_21_30(debug_off, ev_debug_off).
ev_trans_21_30(set_timeout(N), ev_set_timeout(N)).
ev_trans_21_30(halt, ev_halt).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The following are sent only from (pre-3.0) client to facilitator.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ev_trans_21_30(post_event(Event), ev_post_event(NewEvent)) :-
    ev_trans_21_30(Event, NewEvent).
ev_trans_21_30(post_event(To, Event), ev_post_event(To, NewEvent)) :-
    ev_trans_21_30(Event, NewEvent).

ev_trans_21_30(post_query(Goal, Params),
    ev_post_solve(Goal, [reflexive(false) | NewParams])) :-
    params_trans_21_30(Params, NewParams).

% This is the message from a facilitator to its parent facilitator;
% will probably evolve:
% ev_trans_21_30(register_solvable_goals(AGL), register_solvable_goals(AGL)).
% NO, we don't want to translate this. The old form is still handled
% by the new facilitator:
% ev_trans_21_30(register_solvable_goals(GoalList, KSName),
%     ev_register_solvable(add, GoalList, KSName,
%         [if_exists(overwrite)]))].

ev_trans_21_30(solved(GoalId, FromKS, Goal, SolveParams, Solutions),
    ev_solved(GoalId, FromKS, Goal, SolveParams, Solutions)).

/* post_trigger/4: retained for backwards compatibility */
ev_trans_21_30(post_trigger(test, Type, Cond, Action), NewEvent) :-
    ev_trans_21_30(post_trigger(test, Type, unused, unused, Cond, Action),
        NewEvent).

/* post_trigger/4: retained for backwards compatibility */
ev_trans_21_30(post_trigger(data, Type, Cond, Action), NewEvent) :-
    ev_trans_21_30(post_trigger(data, Type,
        [on_write, on_write_replace, on_replace],
        Cond, true, Action), NewEvent).

```

```

/* post_trigger/4: retained for backwards compatibility */
ev_trans_21_30(post_trigger(event, Type, Cond, Action), NewEvent) :-
    ev_trans_21_30(post_trigger(event, Type, [on_receive], Cond, true, Action),
        NewEvent).

ev_trans_21_30(post_trigger(Kind,Recur,OpMask,Template,Test,Action),
    ev_post_trigger_update(add,Mode,Condition,NewAction,Params)) :-
    ( Kind == test -> Mode = task
    | Kind == event -> Mode = comm
    | Kind == alarm -> Mode = time
    | otherwise -> Mode = Kind ),
    ( Recur == whenever ->
        Recurrence = [recurrence(whenever)]
    | otherwise ->
        Recurrence = [recurrence(when)]
    ),
    template_trans_21_30(Kind, Template, Condition),
    ( var(Test) -> TestParam = [] | otherwise -> TestParam = [test(Test)] ),
    ( Mode == data, ev_trans_21_30(Action, NewAction) -> true
    | otherwise -> NewAction = Action ),
    opmask_trans_21_30(OpMask, OpParam),
    ( Mode == data ->
        oaa_Id(FacId),
        Addr = [address(FacId)]
    | otherwise ->
        Addr = []
    ),
    append([Addr, [reply(none), reflexive(false)],
        Recurrence, TestParam, OpParam], Params).
ev_trans_21_30(post_trigger(KS, Kind,Recur,OpMask,Template,Test,Action),
    ev_post_trigger_update(add,Type,Condition,NewAction,Params)) :-
    ( Kind == test -> Type = task
    | Kind == event -> Type = comm
    | Kind == alarm -> Type = time
    | otherwise -> Type = Kind ),
    ( Recur == whenever ->
        Recurrence = recurrence(whenever)
    | otherwise ->
        Recurrence = recurrence(when)
    ),
    template_trans_21_30(Kind, Template, Condition),
    ( var(Test) -> TestParam = [] | otherwise -> TestParam = [test(Test)] ),
    oaa_Id(FacId),
    ( KS == FacId, ev_trans_21_30(Action, NewAction) -> true
    | otherwise -> NewAction = Action ),
    opmask_trans_21_30(OpMask, OpParam),
    append([[address(KS), reply(none), reflexive(false)],
        Recurrence, TestParam, OpParam],
        Params).

params_trans_21_30([], []).
params_trans_21_30([Param | Params], [NewParam | NewParams]) :-
    ( param_trans_21_30(Param, NewParam) ->
        true
    | otherwise ->
        NewParam = Param
    ),

```

```

    params_trans_21_30(Params, NewParams).

param_trans_21_30(cache, cache(true)).
param_trans_21_30(solution_limit(N), solution_limit(N)).
param_trans_21_30(reflexive, reflexive(true)).
param_trans_21_30(address(A), address(NewA)) :-
    ( is_list(A) -> NewA = A | otherwise -> NewA = [A] ).
param_trans_21_30(broadcast, reply(none)).
param_trans_21_30(asynchronous, reply(asynchronous)).
% @@DLM: is this handled?:
param_trans_21_30(test(T), test(T)).
param_trans_21_30(level_limit(N), level_limit(N)).
param_trans_21_30(time_limit(N), time_limit(N)).
% @@DLM: NOT HANDLED!:
param_trans_21_30(and_parallel, and_parallel).
param_trans_21_30(or_parallel, or_parallel).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The following could go to or from the facilitator.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ev_trans_30_21(ev_trace_on, _EvParams, trace_on).
ev_trans_30_21(ev_trace_off, _EvParams, trace_off).
ev_trans_30_21(ev_com_trace_on, _EvParams, tcp_trace_on).
ev_trans_30_21(ev_com_trace_off, _EvParams, tcp_trace_off).
ev_trans_30_21(ev_debug_on, _EvParams, debug_on).
ev_trans_30_21(ev_debug_off, _EvParams, debug_off).
ev_trans_30_21(ev_set_timeout(N), _EvParams, set_timeout(N)).
ev_trans_30_21(ev_halt, _EvParams, halt).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The following are sent only from facilitator to client.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ev_trans_30_21(
    ev_solve(ID, Goal, NewParams),
    _EventParams,
    solve(ID, Goal, Params)) :-
    params_trans_30_21(Params, NewParams).

ev_trans_30_21(ev_reply_solved(_, Solved, Goal, SolveParams, Solutions),
    _EventParams,
    solved(FromKS, Goal, SolveParams, Solutions)) :-
    ( Solved = [FromKS] ->
        true
    | otherwise ->
        FromKS = Solved
    ).

% OBSOLETE: forget these:
% ev_trans_30_21(add_trigger(data, Type, Cond, Action),
% ev_trans_30_21(add_trigger(event, Type, Cond, Action)
% ev_trans_30_21(add_trigger(test, Type, Cond, Action)
% @@DLM: Don't think this is needed:
% ev_trans_30_21(inform_ui(TypeInfo, Result), ))

ev_trans_30_21(

```

```

ev_update_trigger(_ID, add, Type, Condition, Action, TrigParams),
_EventParams,
add_trigger(Kind, Recur, OpMask, Template, Test, Action) ) :-
( Type = task -> Kind == test
| Type = comm-> Kind == event
| Type = time-> Kind == alarm
| otherwise -> Type = Kind ),
( memberchk(recurrence(whenever), TrigParams) ->
Recur = whenever
| otherwise ->
Recur = when
),
Template = Condition,
( memberchk(test(Test), TrigParams) -> true | otherwise -> Test = _ ),
( memberchk(on(OpParam), TrigParams) ->
true
| otherwise ->
OpParam = _
),
opmask_trans_30_21(OpParam, OpMask),
( memberchk(test(Test), TrigParams) -> true | true ).

params_trans_30_21([], []).
params_trans_30_21([Param | Params], [NewParam | NewParams]) :-
( param_trans_30_21(Param, NewParam) ->
true
| otherwise ->
NewParam = Param
),
params_trans_30_21(Params, NewParams).

param_trans_30_21(cache(true), cache).
param_trans_30_21(solution_limit(N), solution_limit(N)).
param_trans_30_21(reflexive(true), reflexive).
% @@DLM: double-check this:
param_trans_30_21(address(A), address(A)).
param_trans_30_21(reply(none), broadcast).
param_trans_30_21(reply(asynchronous), asynchronous).
% @@DLM: is this handled?:
param_trans_30_21(test(T), test(T)).
param_trans_30_21(level_limit(N), level_limit(N)).
param_trans_30_21(time_limit(N), time_limit(N)).
% @@DLM: NOT HANDLED!:
param_trans_30_21(and_parallel, and_parallel).
param_trans_30_21(or_parallel, or_parallel).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% The following are sent only from a pre-3.0 facilitator to a client.
% Backwards compatibility not currently supported.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% ev_trans_21_30(solved(FromKS, Goal, SolveParams, Solutions),
%               ev_reply_solved([FromKS], Solvers, Goal, SolveParams, Solutions)) :-
%   ( Solutions == [] ->
%   Solvers = []
%   | otherwise ->

```

```

%      Solvers = {FromKS}
%      ),
%      ( memberchk(get_address(FromKS), SolveParams) ->
%      true
%      | otherwise ->
%      FromKS = unknown
%      ).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Auxiliary procedures.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Returns either a Singleton list or an empty list.
opmask_trans_21_30(OpMask, []) :-
    var(OpMask),
    !.
opmask_trans_21_30(OpMask, OpParam) :-
    \+ is_list(OpMask),
    !,
    opmask_trans_21_30([OpMask], OpParam).
opmask_trans_21_30([], []).
opmask_trans_21_30([Elt | Rest], [EltTrans | RestTrans]) :-
    opmask_elt_trans_21_30(Elt, EltTrans),
    !,
    opmask_trans_21_30(Rest, RestTrans).
opmask_trans_21_30([_Elt | Rest], RestTrans) :-
    !,
    opmask_trans_21_30(Rest, RestTrans).
opmask_elt_trans_21_30(on_send, on(send)).
opmask_elt_trans_21_30(on_receive, on(receive)).
opmask_elt_trans_21_30(on_write, on(add)).
opmask_elt_trans_21_30(on_retract, on(remove)).
opmask_elt_trans_21_30(on_replace, on(replace)).
% This one probably doesn't have a precise translation:
opmask_elt_trans_21_30(on_write_replace, on(replace)).

opmask_trans_30_21(OpMask, OpMask) :-
    var(OpMask),
    !.
opmask_trans_30_21(OpMask, OpParam) :-
    \+ is_list(OpMask),
    !,
    opmask_trans_30_21([OpMask], OpParam).
opmask_trans_30_21([], []).
opmask_trans_30_21([Elt | Rest], [EltTrans | RestTrans]) :-
    opmask_elt_trans_30_21(Elt, EltTrans),
    !,
    opmask_trans_30_21(Rest, RestTrans).
opmask_trans_30_21([_Elt | Rest], RestTrans) :-
    !,
    opmask_trans_30_21(Rest, RestTrans).
opmask_elt_trans_30_21(on(send), on_send).
opmask_elt_trans_30_21(on(receive), on_receive).
opmask_elt_trans_30_21(on(add), on_write).
opmask_elt_trans_30_21(on(remove), on_retract).
opmask_elt_trans_30_21(on(replace), on_replace).
% This one probably doesn't have a precise translation:

```

```

opmask_elt_trans_30_21(on(replace), on_write_replace).

template_trans_21_30(data,
                    data(ksdata, [AgentId,Status,Solvables,Name]),
                    agent_data(AgentId,Status,Solvables,Name)) :-
    !.
template_trans_21_30(data, Template, Template) :-
    !.
template_trans_21_30(event, Template, Condition) :-
    !,
    ev_trans_21_30(Template, Condition).
template_trans_21_30(_, Template, Template).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Event handlers for selected pre-3.0 events.
%
% In these cases, this approach is easier than providing an event
% translation.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

oaa_AppDoEvent(register_solvable_goals(GoalList), Params) :-
    memberchk( connection_id(Connection), Params),
    % This hack inherited from b.pl:
    oaa_AppDoEvent(register_solvable_goals(GoalList, Connection),
                    Params).

oaa_AppDoEvent(register_solvable_goals(GoalList, Name), Params) :-
    memberchk( connection_id(Connection), Params),
    update_connected(Connection, [oaa_name(Name)]),
    icl_ConvertSolvables(GoalList, Solvables),
    oaa_AppDoEvent(ev_register_solvables(add,Solvables,Name,[if_exists(overwri
te)]),
                    Params).

oaa_AppDoEvent(can_solve(Goal), EvParams) :-
    memberchk(from(KS), EvParams),
    findall(SomeKS, choose_ks_for_goal(KS, Goal, _, [], SomeKS, _), AgentList),
    oaa_PostEvent(return_can_solve(Goal, AgentList), [address(KS)]).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% BB events
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

oaa_AppDoEvent(write_bb(ksdata, [Id,Status,Solvables,Name]),
                EvParams) :-
    !,
    ( var(Solvables) ->
        % (Surely this never happens.)
        oaa:oaa_add_data_local(agent_data(Id,Status,Solvables,Name), [from(Id)])
    | otherwise ->
        icl_ConvertSolvables(Solvables, FormalSolvables),
        oaa_AppDoEvent(ev_register_solvables(add,FormalSolvables,Name,
                                                [if_exists(overwrite)]),
                        [from(Id) | EvParams])
    ).
oaa_AppDoEvent(write_bb(oaa_version, V), EvParams) :-

```



```

!,
memberchk(from(Id), EvParams),
% oaa: oaa_add_data_local(data(oaa_Version, V), [from(Id)]),
com_GetInfo(ConnectionId, oaa_id(Id)),
com_AddInfo(ConnectionId, agent_version(V)).
oaa_AppDoEvent(write_bb(language, Language), EvParams) :-
!,
memberchk(from(Id), EvParams),
com_GetInfo(ConnectionId, oaa_id(Id)),
com_AddInfo(ConnectionId, agent_language(Language)).
oaa_AppDoEvent(write_bb(kshost, Host), EvParams) :-
!,
memberchk(from(Id), EvParams),
oaa: oaa_solve_local(agent_data(Id, _, _, Name), []),
oaa: oaa_add_data_local(agent_host(Id, Name, Host),
                        [from(Id) | EvParams]).
oaa_AppDoEvent(write_bb(Item, Data), EvParams) :-
!,
memberchk(from(Id), EvParams),
oaa: oaa_add_data_local(data(Item, Data), [from(Id)]).

oaa_AppDoEvent(write_once_bb(Item, Data), EvParams) :-
(Item = ksdata ; Item = oaa_version ; Item = language ; Item = kshost),
!,
oaa_AppDoEvent(write_bb(Item, Data), [single_value(true) | EvParams]).
oaa_AppDoEvent(write_once_bb(Item, Data), EvParams) :-
!,
memberchk(from(Id), EvParams),
oaa: oaa_add_data_local(data(Item, Data), [from(Id), single_value(true)]).

oaa_AppDoEvent(write_replace_bb(Item, Data), EvParams) :-
(Item = ksdata ; Item = oaa_version ; Item = language ; Item = kshost),
!,
oaa_AppDoEvent(write_bb(Item, Data), [unique_values(true) | EvParams]).
oaa_AppDoEvent(write_replace_bb(Item, Data), EvParams) :-
!,
memberchk(from(Id), EvParams),
oaa: oaa_add_data_local(data(Item, Data), [from(Id), unique_values(true)]).

oaa_AppDoEvent(replace_bb(ksdata, [A,open,C,Name], [A,ready,C,Name]),
                EvParams) :-
!,
oaa_AppDoEvent(ev_ready(Name), EvParams).
oaa_AppDoEvent(replace_bb(ksdata, [Id,Status,Solvables,Name],
                          [NewId,NewStatus,NewSolvables,NewName]),
                EvParams) :-
!,
( var(NewSolvables) ->
  oaa: oaa_replace_data_local(agent_data(Id,Status,Solvables,Name),
    [from(Id), with(agent_data(NewId,NewStatus,NewSolvables,NewName))])
| otherwise ->
  ' icl_ConvertSolvables(NewSolvables, FormalSolvables),
  oaa_AppDoEvent(ev_register_solvables(add,FormalSolvables,NewName,
    [if_exists(overwrite)]),
    [from(NewId) | EvParams])
).
oaa_AppDoEvent(replace_bb(Item,OldData,NewData), EvParams) :-

```

```

!,
memberchk(from(Id), EvParams),
oaa:oaa_replace_data_local(data(Item,OldData),
    [from(Id), with(data(Item,NewData))]).

% @@DLM: May need more special-purpose clauses starting here:
oaa_AppDoEvent(retract_bb(Item,Data), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    oaa:oaa_remove_data_local(data(Item,Data), [from(Id)]).

oaa_AppDoEvent(read_bb(ksdata,[AgentId,Status,Solvables,Name]), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    findall(read_bb(ksdata,[AgentId,Status,Solvables,Name]),
        oaa:oaa_solve_local(agent_data(AgentId,Status,Solvables,Name), []),
        Solutions),
    oaa_simplify_ksdata(Solutions, Simplified),
    oaa_PostEvent(return_read_bb(Simplified), [address(Id)]).
oaa_AppDoEvent(read_bb(KS,kshost,Host), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    findall(read_bb(KS,kshost,Host),
        oaa:oaa_solve_local(agent_host(KS,_,Host), []),
        Solutions),
    oaa_PostEvent(return_read_bb(Solutions), [address(Id)]).
oaa_AppDoEvent(read_bb(oaa_version,V), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    % Not sure if this works (but this clause is probably never called):
    findall(read_bb(oaa_version,V),
        ( com_GetInfo(ConnectionId, oaa_id(_)),
          com_GetInfo(ConnectionId, agent_version(V)) ),
        Solutions),
    oaa_PostEvent(return_read_bb(Solutions), [address(Id)]).

oaa_AppDoEvent(read_bb(KS,oaa_version,V), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    findall(read_bb(KS,oaa_version,V),
        ( com_GetInfo(ConnectionId, oaa_id(KS)),
          com_GetInfo(ConnectionId, agent_version(V)) ),
        Solutions),
    oaa_PostEvent(return_read_bb(Solutions), [address(Id)]).
oaa_AppDoEvent(read_bb(Item,Data), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    findall(read_bb(Item,Data),
        oaa:oaa_solve_local(data(Item,Data), []),
        Solutions),
    oaa_PostEvent(return_read_bb(Solutions), [address(Id)]).
% @@The owner parameter isn't implemented yet for solve!
oaa_AppDoEvent(read_bb(_KS,Item,Data), EvParams) :-
    !,
    memberchk(from(Id), EvParams),
    findall(read_bb(Item,Data),
        oaa:oaa_solve_local(data(Item,Data), []),

```

```

        Solutions),
    oaa_PostEvent(return_read_bb(Solutions), [address(Id)]).

oaa_simplify_ksdata([], []).
oaa_simplify_ksdata([KSDData | Rest], [Simplified | RestSimp]) :-
    KSDData = read_bb(ksdata, [A, B, Solvables, D]),
    icl_ConvertSolvables(SimplifiedSolvables, Solvables),
    Simplified = read_bb(ksdata, [A, B, SimplifiedSolvables, D]),
    oaa_simplify_ksdata(Rest, RestSimp).

```